(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平11－272466

(43)公開日　平成11年(1999)10月８日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　9/38	３１０	Ｇ０６Ｆ　9/38	３１０Ｆ
	３５０		３５０Ｂ
			３５０Ａ

審査請求　未請求　請求項の数７　ＯＬ　（全 11 頁）

(21)出願番号　特願平11－25857

(22)出願日　平成11年(1999)２月３日

(31)優先権主張番号　０９／０２１１３４

(32)優先日　1998年２月10日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
INTERNATIONAL BUSINESS MASCHINES CORPORATION
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　シーシウン・スティーブン・タン
アメリカ合衆国78726　テキサス州オースチン　パープルック・ドライブ　9923

(74)代理人　弁理士　坂口　博　（外１名）

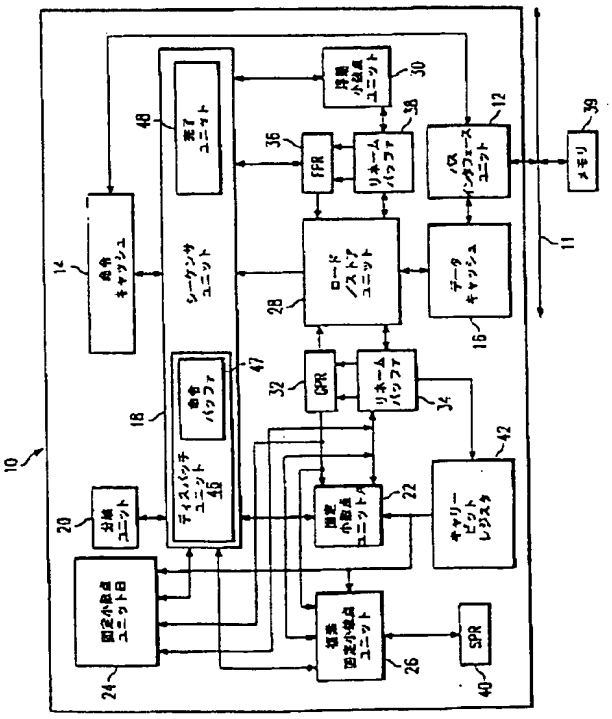
最終頁に続く

(54)【発明の名称】　ロード／ロード検出及びリオーダー方法及び装置

(57)【要約】

【課題】　本発明は、データ・コヒーレンシを侵害する可能性のあるアウト・オブ・オーダー命令を検出してリオーダーする手段に関する。

【解決手段】　本発明は、各項目がコンピュータ・マイクロプロセッサ内の命令に対応する命令データの項目を保持するミスキュー・テーブルを含む。各項目内の命令データは、ｉ）命令のアドレス情報と、ii）ミスキュー・テーブル内の他の命令を基準にした命令の順序を示す命令の順序情報と、iii）データが変更された可能性を示す、命令のデータ変更情報と、iv）前記項目に対応するより古い命令より前により新しい命令が完了したことを示すアウト・オブ・オーダー情報とを含む。本発明は、完了した命令のアドレスをミスキュー内のアドレス情報項目と比較するアウト・オブ・オーダー比較機構も含む。



1

【特許請求の範囲】

【請求項１】命令のアウト・オブ・オーダー実行のための装置であって、

ａ）各項目がコンピュータ・マイクロプロセッサ内の命令に対応する命令データの項目を保持するミスキュー・テーブルを含み、前記命令データは、

ｉ）命令のアドレス情報と、

ii）他の命令を基準にした前記対応する命令の順序を示す命令の順序情報と、

iii）アドレス情報フィールドに対応するアドレスにあるデータを使用するより新しい命令が現行項目より前に完了したことを示すアウト・オブ・オーダー情報と、

iv）前記アドレス情報フィールドに対応する前記アドレスにあるデータが変更された可能性を示す、命令のデータ変更情報とを含み、

前記装置は、

ｂ）比較された項目内のアドレス情報フィールドに対応するアドレスにあるデータを使用する、より新しい命令が前記比較された項目より前に完了した場合、前記比較された項目のアウト・オブ・オーダー情報フィールドが設定される、前記ミスキュー・テーブル内の項目に前記アウト・オブ・オーダー情報フィールドを設定するアウト・オブ・オーダー比較機構と、

ｃ）前記ミスキュー・テーブル内の前記アドレス情報フィールド内のアドレス情報を、変更された可能性のあるアドレスと比較する変更比較機構とを含み、前記変更されたアドレスが比較された前記命令項目内の前記アドレス情報と一致する場合、前記項目内の前記変更フィールドが、前記アドレスにある変更されたデータを示すようにマークされる装置。

【請求項２】アウト・オブ・オーダー命令と変更されたデータとを示す命令項目に対応する命令のすべての後続命令を取り消す、請求項１に記載の装置。

【請求項３】前記アウト・オブ・オーダー比較機構が項目内のすべてのデータ・アドレス情報を比較して前記アドレスが一致するかどうかを判断する、請求項１に記載の装置。

【請求項４】前記アウト・オブ・オーダー比較機構が、項目内の前記データ・アドレス情報の一部を比較して、前記アドレスが一致するかどうかを判断する、請求項１に記載の装置。

【請求項５】データ・コヒーレンシ侵害を生じさせる可能性のあるマイクロプロセッサ内のアウト・オブ・オーダー命令を検出する方法であって、

ａ）データ・アドレスを有する前記マイクロプロセッサ上で実行する新規命令を作成するステップと、

ｂ）前記新規命令が実行されるように設定されている場合、

ｉ）前記命令のデータ・アドレスをミスキュー・テーブル内の、前の命令に対応し、アドレス情報と命令順序情報とアウト・オブ・オーダー標識と変更データ標識とを含む既存の命令項目と比較し、

ii）前記新規命令のデータ・アドレスがミスキュー・テーブル内の項目のアドレス情報と一致する場合、前記ミスキュー・テーブル内の前記一致項目をアウト・オブ・オーダー命令にするステップと、

ｃ）前記新規命令が実行されるように設定されていない場合、前記ミスキュー・テーブル内の前記新規命令の項目を作成し、前記新規命令のデータ・アドレスが前記新規命令項目のアドレス情報に入れられ、前記命令項目の順序情報に前記命令の順序に関する情報が入れられる方法。

【請求項６】ｄ）前記ミスキュー・テーブルを絶えず走査して、実行するように設定されている命令に対応する項目がないか調べるステップと、

ｅ）前記ミスキュー・テーブル内の項目に対応する命令が実行されるように設定されている場合、

ｉ）実行するように設定されている前記命令に対応する前記項目のアドレス情報を、実行するように設定されている前記命令より古い命令に対応する前記ミスキュー・テーブル内の項目のアドレス情報と比較するステップと、

ii）前記ミスキュー・テーブル内に、実行するように設定されている前記命令より古い命令に対応する項目が見つかった場合、及び実行するように設定されている前記命令に対応する前記項目のアドレス情報が前記より古い命令に対応する前記項目内のアドレス情報と一致する場合、前記より古い命令に対応する前記項目をアウト・オブ・オーダーとしてマークするステップと、

iii）実行するように設定されている前記命令に対応する前記項目を前記ミスキュー・テーブルから除去するステップとをさらに含む、請求項５に記載の方法。

【請求項７】ｆ）データ変更事象がないか絶えず走査するステップと、

ｇ）データ変更事象が発生した場合、

ｉ）前記変更されたデータのアドレスを前記ミスキュー・テーブルにブロードキャストするステップと、

ii）前記変更されたデータのアドレスを前記ミスキュー・テーブルの項目内のアドレス情報と比較するステップと、

iii）前記変更されたデータのアドレスと前記ミスキュー・テーブル内の項目のアドレスが一致する場合、前記一致するアドレスを有する前記ミスキュー・テーブル内の項目を変更されたものとしてマークするステップとをさらに含む、請求項６に記載の方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は一般にはコンピュータ・プロセッサの分野に関し、より詳細には、単一マイクロプロセッサ・チップ上に集積されたプロセッサに関

する。さらに詳細には、本発明は、特に複数ＣＰＵシステムにおける、アウト・オブ・オーダー処理に内在するデータ・コヒーレンシ問題の検出と是正に関する。

【０００２】

【従来の技術】マイクロプロセッサの一層の高速化は、現在のプロセッサ設計の主要な目標の１つである。プロセッサのパフォーマンスを向上させるために多くの異なる技法が採用されてきた。プロセッサ・パフォーマンスを大幅に向上させる１つの技法は、キャッシュ・メモリの使用である。本明細書では、キャッシュ・メモリとは、マイクロプロセッサ自体の上に形成されており、その結果、マイクロプロセッサ・チップから離れた位置にあるＲＡＭや磁気ディスクなどの他のタイプのメモリよりもアクセス時間がはるかに高速な１組の記憶場所を指す。頻繁に使用されるデータのコピーをキャッシュに記憶することによって、プロセッサはそのデータが必要なときにキャッシュにアクセスすることができ、その情報を入手するために「オフ・チップ（チップ外）」まで行く必要がなく、その結果、プロセッサのパフォーマンスが大幅に強化される。

【０００３】しかし、キャッシュ・メモリには特定の問題が付随する。具体的には、１つのシステムで複数のプロセッサが使用され、同じデータを必要とする場合に大きな問題がある。この場合、システムは要求されるデータが整合するように、すなわちその時点でそのプロセッサにとって有効であるように保証する必要がある。データが１つのプロセッサのキャッシュに記憶されており、別のプロセッサがそれと同じ情報を要求する場合にも問題がある。

【０００４】スーパースカラ・プロセッサは命令を非プログラム順序（アウト・オブ・オーダー）で実行することができるため、従来のスカラ・プロセッサより有利なパフォーマンスを実現する。この方法により、実行速度の遅い命令を保留にしている間にプロセッサ上の他の資源を使用して実行することができる後続命令が、実行の遅い方の命令によって停止することがなくなる。

【０００５】典型的なアーキテクチャでは、すべての命令が１つのデータを必要とする場合、プロセッサはまずオンボード・キャッシュに行って、そのデータがオンボード・キャッシュに入っていないかどうか調べる。キャッシュによっては、２つの外部ポートを有するものがあり、キャッシュをインタリーブさせることができる。これは、たとえば図１で、キャッシュ１００が２つのキャッシュ・バンク１４０及び１３０を有することを意味する。一方のキャッシュ・バンクは奇数アドレス用とすることができ、他方のキャッシュ・バンクは偶数アドレス用となる。

【０００６】内部的には、各キャッシュ・バンク１４０及び１３０キャッシュが内部入力ポート（図示せず）を有し、このポートに対してキャッシュ要求のアドレス情報が入力される。図１で、アドレスＡ１のデータはキャッシュ・バンク１３０内のキャッシュ・ライン１１０に記憶され、アドレスＡ２のデータはキャッシュ・バンク１４０内のキャッシュ・ライン１２０に記憶される。キャッシュ１００は入力データ用の２つの外部ポートであるポート１８０とポート１９０を有する。

【０００７】キャッシュ要求１は、命令１（図示せず）のキャッシュ要求を示し、要求２は命令２（図示せず）のキャッシュ要求を示す。命令１は命令２より古い命令であり、これは命令２よりも前に実行する必要があることを意味する。米国テキサス州オースチンのＩＢＭコーポレイションのＰｏｗｅｒＰＣ（ＴＭ）プロセッサのようにスーパースカラ・プロセッサが複数のロード・ユニットを有する場合、両方の命令が同時にキャッシュ要求を行うことができる。図の例では、命令２と命令１がアドレスＡ１にあるデータにアクセスを試みようとして、キャッシュ１００にそれを行う要求を渡している。

【０００８】バンク１３０には内部入力ポートが１つしかないため、同時に両方のキャッシュ要求を処理することができない。これは、キャッシュ１００のインタリーブ性によるものである。

【０００９】図２に、キャッシュ要求２がキャッシュ要求１よりも前にキャッシュ・バンク１３０にアクセスするとどうなるかを示す。キャッシュ要求２はキャッシュ・バンク１３０で必要なデータにヒットする。しかし、キャッシュ要求１は少なくとも次のサイクルまでキャッシュ・バンク１３０にアクセスすることができない。したがって、より新しい命令２は必要なデータを、より古い命令１が入手するよりも前に入手することができる。この場合、このポート割り振り競合のために、この場合、より新しい命令２がより古い命令１より前に完了することができる。

【００１０】これと同じ順序づけ問題は、より古い命令がキャッシュでミスし、より新しい命令がヒットした場合にも起こり得る。ミスは、データのアドレスがメモリ管理ユニット内に見つからない場合に起こり、その場合、メモリ管理ユニットはそのデータをより上位のメモリから取り出すように要求しなければならない。ヒットは、データのアドレスとデータの両方に、メモリ管理ユニットとキャッシュを介してアクセスすることができる場合に起こり、そのデータはそれを待っている命令に出力することができる。

【００１１】データの実アドレスが２つの異なる有効アドレスによって表されている場合、両方の命令が同じデータにアクセスを試み、より古い命令のキャッシュ・ミスの後により新しい命令によるキャッシュ・ヒットがあるという状況が発生する。より新しい命令によって要求された有効アドレスとそのデータがすでにメモリ管理ユニットとキャッシュによってアクセス可能な場合であって、メモリ管理ユニットとキャッシュでより古い命令の

アドレスとデータにアクセスすることができない場合にも、より古い命令と同じデータにアクセスするより新しい命令がより古い命令より前に完了することができるという状況が起こる。

【００１２】マルチプロセッサ・システムでは、１つのプロセッサでのキャッシュ・ミスによって、システム内の他のプロセッサに対する「スヌープ」要求をトリガすることができる。このスヌープ要求は、他のプロセッサに対して、「スヌープ」されているデータが別のプロセッサによって要求されていることを示し、他のプロセッサは求められているアドレスがそれぞれのプロセッサ自体のキャッシュに入っているかどうかを調べるように指示する。プロセッサのキャッシュに入っている場合、メイン・メモリ・データを整合させる必要がある。すなわち、システム状態の正しい現在の状態を反映するように更新する必要がある。

【００１３】スーパースカラ・アーキテクチャに関しては、ロード命令がアウト・オブ・オーダーで終了することがあるため、言い換えると、新しい命令が古い命令よりも前に完了としてマークされることがあるため、この問題はさらに大きくなる。すなわち、新しい命令が古い命令よりも前に実行されるように設定されているものとマークされることがある。したがって、２つのロード命令が同じキャッシュ場所をアドレス指定し、実際には１つのデータが古い命令より前に新しい命令に供給されることがある。したがって、新しい命令は、アウト・オブ・オーダーで完了するマークが付けられ、それによってその命令の完了に不正なデータが使用される可能性がある。後続のロード命令がより古いロード命令を迂回する場合、より古いロード命令は、元のプログラム順序に基づいて受け取るはずだったデータよりも新しいデータを入手する可能性がある。

【００１４】このコヒーレンシ問題の従来の解決策としては、ロード・キューにページ索引と実アドレスを、ＩＤ及び有効ビットと共に保持する。ＩＤはロード命令のプログラム順序を示す。

【００１５】上述の項目に加えて、ロード・キュー項目は、そのアドレスのキャッシュ・ライン項目が変更されたかどうかを示す変更フィールドも保持する。ストア命令やスヌープ要求などのキャッシュ・アクセスによって、キャッシュ・ラインが変更されたことが示されると、ロード・キューが探索される。それと同じラインの項目が含まれている場合、変更の可能性を示すように変更ビットが設定される。

【００１６】後続のロードは、ロード・キュー項目の比較を行うことになる。ロード・キュー内で同じラインが保留中であり、変更済みとマークされている場合、ＩＤフィールドを検査する。現在のラインが保留中だったラインよりも古く、変更されている場合、ロード・キュー内の保留ロードが取り消され、その後続ロードの後に再

実行される。これによって、より古いロードがより新しいロードよりも新しいデータを持つという問題が回避される。

【００１７】

【発明が解決しようとする課題】

【課題を解決するための手段】本発明は、ロード・キューをなくし、不正なデータを使用して終了した可能性がある命令を取り消すだけの新規な手段を提供する。

【００１８】本発明は、拒否されたキャッシュへのアクセスの試行や、命令が完了できないその他の理由を保持するミスキュー・テーブルを提供する。

【００１９】好ましい実施形態では、すべての命令がミスキュー・テーブル内に初期項目を作成する。その命令のデータがキャッシュに入っており、入手可能な場合、ミスキュー・テーブル内のその命令項目がミスキュー・テーブルから取り出される。これは、プロセッサが、順序づけ問題の可能性がないかミスキュー・テーブルを探索してから行う。次に、その命令には、プロセッサのシーケンシング・ユニットによって、キャッシュ内で見つかったデータを使用して完了のマークが付けられる。

【００２０】古い命令が新しい命令の後に完了し、新しい命令と古い命令が同じアドレスにあるデータにアクセスする場合、順序づけ問題が起こる。

【００２１】命令のデータをキャッシュで入手できない場合、あるいは他の何らかの理由で完了できない場合、ミスキュー・テーブル内に作成された命令項目は、命令が完了可能になるまでミスキュー・テーブルにとどまる。キャッシュ内でデータを入手できるようになった場合など、命令が完了可能になると、命令には完了のマークが付けられる。次に、命令項目はミスキュー・テーブルから削除される。

【００２２】命令が完了可能になると、それに付随する項目がミスキュー・テーブルから削除される。ミスキュー・テーブルで、同じアドレスを持つ、前の命令項目がないか探索を行う。完了する項目と同じアドレスを持つ前の命令項目が見つかった場合、前の命令項目はまだ完了していないため、アウト・オブ・オーダーとしてマークされる。前述のように、これはたとえば、命令が同じデータ・アドレスに別名を付け、より古い命令にデータが使用可能になっていることがキャッシュから通知されていない場合に起こることがある。

【００２３】すべての命令がミスキュー・テーブルに項目を作成して完了可能な命令項目が次のサイクルで引き出されるか、拒否された命令のみがミスキュー・テーブルに項目を作成するかを問わず、機能上の結果は同じであることに留意されたい。この結果とは、ただちに完了する命令に対応する命令項目はミスキュー・テーブルにはとどまらず、ただちには完了しない命令に対応する命令項目のみがミスキュー・テーブル内に残るということである。

【００２４】これによって、すべての現行命令が有効であって実行するように設定されていると見なされるか、データを待ってミスキュー・テーブルに入れられている対応する項目を有するとみなされるため、ロード・キューを備える必要がなくなる。

【００２５】現行命令が完了した場合、データのアドレスをミスキュー・テーブル内の項目と照合して一致があるかどうかを調べる。一致するものがある場合、ミスキュー・テーブル内の一致項目がアウト・オブ・オーダーとしてマークされる。すなわち、アドレスからデータにアクセスする、より新しい命令が、同じアドレスに対する古い命令よりも先に完了するように設定されており、次に、より古い命令がアウト・オブ・オーダーで完了し、ミスキュー・テーブル内のその項目がそれに従ってマークされることになる。

【００２６】さらに、キャッシュがミスキュー・テーブルに暫時、項目を有する命令のために有効なデータを返すと、その項目に対応する命令が完了に設定される。そのアドレスにあるデータにアクセスするまだ完了していない命令に対応する古い命令項目についても、ミスキュー・テーブルで同様の探索が行われる。一致する物が見つかると、一致する古い項目はアウト・オブ・オーダーとしてマークされる。

【００２７】スヌープ要求など、データ・コヒーレンシ変更事象が発生した場合、ミスキュー・テーブルに問い合わせる。データ・コヒーレンシ変更事象と同じアドレスを持つ項目には変更済みのマークが付けられる。

【００２８】ミスキュー・テーブルから命令項目が解放されると、命令を完了する。プロセッサは次に、特定の事象が発生したかどうかを判断する。ミスキュー・テーブル内の命令項目によって、その項目に対応する命令がアウト・オブ・オーダーであってしかも変更されている場合、そのアウト・オブ・オーダーかつ変更済み命令の後に実行されると思われる命令は取り消され、そして再実行され、したがってデータ・コヒーレンシが保たれる。

【００２９】

【発明の実施の形態】図３は、本発明により情報を処理するプロセッサ・システム１０を示すブロック図である。好ましい実施形態では、プロセッサ１０は米国テキサス州オースチンのＩＢＭコーポレイションのＰｏｗｅｒＰＣ™プロセッサなどの単一集積回路スーパースカラ・マイクロプロセッサである。したがって、後で詳述するように、プロセッサ１０は様々なユニット、レジスタ、バッファ、メモリ、及びその他の部分を含み、それらはすべて集積回路によって形成されている。また、好ましい実施形態では、プロセッサ１０は縮小命令セット・コンピューティング（「ＲＩＳＣ」）技法に従って動作する。図３に示すように、システム・バス１１がプロセッサ１０のバス・インタフェース・ユニット（「ＢＩＵ」）１２に接続されている。ＢＩＵ１２は、プロセッサ１０とシステム・バス１１との間の情報の伝送を制御する。

【００３０】ＢＩＵ１２は、プロセッサ１０の命令キャッシュ１４とデータ・キャッシュ１６とに接続されている。命令キャッシュ１４は、シーケンサ・ユニット１８に命令を出力する。シーケンサ・ユニット１８は、命令キャッシュ１４からのこのような命令に応答して、プロセッサ１０の他の実行回路に命令を選択的に出力する。

【００３１】好ましい実施形態では、プロセッサ１０の実行回路は、ディスパッチ・ユニット４６と完了ユニット４８の実行ユニットを含むシーケンサ・ユニット１８に加えて、複数の実行ユニット、すなわち分岐ユニット２０と、固定小数点ユニットＡ（「ＦＸＵＡ」）２２と、固定小数点ユニットＢ（「ＦＸＵＢ」）２４と、複素固定小数点ユニット（「ＣＦＸＵ」）２６と、ロード／ストア・ユニット（「ＬＳＵ」）２８と、浮動小数点ユニット（「ＦＰＵ」）３０とを含む。ＦＸＵＡ２２、ＦＸＵＢ２４、ＣＦＸＵ２６、及びＬＳＵ２８は、それぞれのソース・オペランド情報を汎用アーキテクチャ・レジスタ（「ＧＰＲ」）３２及び固定小数点リネーム・バッファ３４から入力する。さらに、ＦＸＵＡ２２とＦＸＵＢ２４は、キャリー・ビット（「ＣＡ」）レジスタ４２から「キャリー・ビット」を入力する。ＦＸＵＡ２２、ＦＸＵＢ２４、ＣＦＸＵ２６、及びＬＳＵ２８は、それぞれの演算の結果（目的オペランド情報）を出力して、固定小数点リネーム・バッファ３４に記憶する。また、ＣＦＸＵ２６は、特殊目的レジスタ（「ＳＰＲ」）４０との間でソース・オペランド情報と目的オペランド情報を入出力する。

【００３２】ＦＰＵ３０は、そのソース・オペランド情報を浮動小数点アーキテクチャ・レジスタ（「ＦＰＲ」）３６と浮動小数点リネーム・バッファ３８から入力する。ＦＰＵ３０は演算の結果（目的オペランド情報）を出力して浮動小数点リネーム・バッファ３８内の選択された項目に記憶する。

【００３３】シーケンサ・ユニット１８は、ＧＰＲ３２及びＦＰＲ３６との間で情報を入出力する。シーケンサ・ユニット１８から、分岐ユニット２０が命令と、プロセッサ１０の現在の状態を示す信号とを入力する。分岐ユニット２０は、このような命令及び信号に応答して、プロセッサ１０によって実行される命令のシーケンスを記憶する適切なメモリ・アドレスを示す信号を（シーケンサ・ユニット１８に）出力する。シーケンサ・ユニット１８は、分岐ユニット２０からのこのような信号に応答して、命令キャッシュ１４からの指示された命令シーケンスを入力する。命令シーケンスのうちの１つまたは複数の命令が命令キャッシュ１４に記憶されていない場合、命令キャッシュ１４はシステム・バス１１に接続されているシステム・メモリ３９から（ＢＩＵ１２とシス

テム・バス１１を介して）その命令を入力する。

【００３４】シーケンサ・ユニット１８は命令キャッシュ１４からの命令入力に応答して、ディスパッチ・ユニット４６を介して、実行ユニット２０、２２、２４、２６、２８、及び３０のうちから選択された実行ユニットに選択的にディスパッチする。各実行ユニットは、特定の命令クラスの１つまたは複数の命令を実行する。たとえば、ＦＸＵＡ２２とＦＸＵＢ２４はソース・オペランドに対して加算や減算、ＡＮＤ、ＯＲ、ＸＯＲなどの第１のクラスの固定小数点数値演算を実行する。ＣＦＸＵ２０はソース・オペランドに対して固定小数点乗算や除算など第２のクラスの固定小数点演算を実行する。ＦＰＵ３０は、ソース・オペランドに対して浮動小数点乗算や除算などの浮動小数点演算を実行する。

【００３５】プロセッサ１０は、実行ユニット２０、２２、２４、２６、２８、及び３０のうちの様々な実行ユニットで同時に複数の命令を処理することによって高パフォーマンスを実現する。したがって、各命令はいくつかの段階から成るシーケンスとして処理され、各段階は他の命令の段階と並列して実行可能である。このような技術は「パイプライン処理」と呼ばれる。好ましい実施形態の重要な態様では、命令は通常６段階、すなわち、フェッチ、デコード、ディスパッチ、実行、完了、及びライトバックの各段階で処理される。

【００３６】好ましい実施形態では、各命令は命令処理の各段階を完了するのに１マシン・サイクルを必要とする。それにもかかわらず、ある種の命令（たとえばＣＦＸＵ２６によって実行される複素固定小数点命令）は、複数のサイクルを必要とすることがある。したがって、特定の命令の実行と段階と完了段階との間に、先行命令の完了に要する時間の変動に応じて変動する遅延が生じることがある。

【００３７】ＬＳＵ２８は、ロード命令に応答して、データ・キャッシュ１６から情報を入力し、その情報をリネーム・バッファ３４及び３８のうちの選択されたリネーム・バッファにコピーする。その情報がデータ・キャッシュ１６に記憶されていない場合、データ・キャッシュ１６はその情報をシステム・バス１１に接続されているシステム・メモリ３９から（ＢＩＵ１２とシステム・バス１１を介して）入力する。さらに、データ・キャッシュ１６は、データ・キャッシュ１６からシステム・バス１１に接続されているシステム・メモリ３９に（ＢＩＵ１２とシステム・バス１１）を介して情報を出力することができる。

【００３８】図４を参照すると、本発明の一実施形態によりロード命令などの命令を処理する回路を図示する略図が示されている。制御論理回路を含むデータ・ユニット２０４に対するアドレスは、キャッシュ２０６への物理的アクセスを必要とする。キャッシュ２０６は、データがキャッシュに入っている場合、処理のためにキャッシュ２０６からフォーマッタ２１０にデータを渡す、この例では６４ビットのデータ・ラインに接続された出力ポートを有する。

【００３９】本発明の一実施形態では、命令がディスパッチされるたびにミスキュー・テーブル６００内に項目が作成される。命令がデータ・キャッシュ内でヒットした場合、次のサイクルでその命令の項目がミスキュー・テーブル６００から除去される。しかし、命令がデータ・キャッシュ内でミスした場合は、その実アドレスと、場合によっては有効アドレスと、その他の情報がミスキュー・テーブル６００に残る。プロセッサはミスキュー・テーブル内の項目のアドレス情報を絶えず走査し、各サイクルで、プロセッサはミスキュー・テーブル６００に記憶されている有効アドレスにあるキャッシュにアクセスを試みる。最終的に、ミスキュー・テーブル６００内の各項目についてキャッシュ内でデータが入手可能になり、処理のためにフォーマッタに渡される。

【００４０】上述のように有効アドレスを介してキャッシュにアクセスする代わりに、マイクロプロセッサはキャッシュに記憶されている実アドレスを介してキャッシュにアクセスを試みることもできることに留意されたい。これは、実施上の問題であり、本発明全体には影響しないことに留意されたい。

【００４１】しかし、現行命令を最初からミスキュー・テーブル６００に入れなくても本発明は機能することに留意されたい。現行命令は、理由を問わず、シーケンシング・ユニットによって最初に導入された直後に完了するものとして設定することができない場合にのみ、ミスキュー・テーブル６００で示す必要がある。

【００４２】好ましい実施形態では、命令項目はミスキュー・テーブル６００にイン・オーダーで記憶される。これを図５に示す。命令項目４１０を生成する命令は命令項目４２０を生成する命令よりも古く、したがって命令項目４１０はミスキュー・テーブル６００内で命令４２０よりも上位に記憶される。しかし、適切な識別情報と順序づけ情報があれば、ミスキュー・テーブルに命令を順不同で記憶することもできることに留意されたい。

【００４３】ミスキュー・テーブル内の項目を図６に示す。本発明に必要な最低限の情報は、アドレス情報フィールド５１０と、アウト・オブ・オーダー情報フィールド５２０と、変更データ情報フィールド５３０である。アドレス情報は、実アドレス情報５４０と有効アドレス情報５５０を含むサブフィールドを有することもできる。ミスキュー・テーブル項目は、有効フィールド５６０や命令ＩＤフィールド５７０など、それに付随するその他の情報を有することもでき、それらは順序づけ情報として使用できる。

【００４４】他の実施形態では、ミスキュー・テーブル内の項目はアウト・オブ・オーダーでミスキュー・テーブルに記憶される。図６の有効フィールド５６０は、ミ

スキュー・テーブル項目が実際にまだミスキュー・テーブル６００に入っているかどうかを示す。新しい項目は、有効フィールド５６０が設定されていない最初のミスキュー・テーブル行に作成される。命令ＩＤフィールド５７０を使用して順序づけ情報が保持される。

【００４５】次に図７を参照すると、プロセッサによって最初に、アドレスＤにあるデータにアクセスする新規の命令８００が渡される。好ましい実施形態では、ミスキュー・テーブル６００内の次に使用可能なスロットに新規命令８００の項目６１０が作成される。新規命令８００はただちに完了し、プロセッサはミスキュー・テーブル６００内のより古い項目６２０、６３０、及び６４０を新規命令８００と照合する。具体的には、新規命令８００のアドレスを、ミスキュー・テーブル６００内のすべてのより古い項目のアドレス情報５１０と照合して一致するアドレスがないか調べる。当然ながら、ミスキュー・テーブル６００内に含まれるすべての項目は新規命令８００より古い命令に対応していなければならない。

【００４６】新規命令８００は次のサイクルで完了し、項目６１０が削除されることに留意されたい。また、項目６１０はただちに作成される必要はないことにも留意されたい。新規命令８００のオペランド・アドレスをミスキュー・テーブル６００の項目内のアドレス情報と照合し、新規命令８００が次のクロック・サイクルで完了しない場合にのみ、その命令の項目６１０が作成されることになる。

【００４７】代替実施形態では、項目６１０は削除されず、有効ミスキュー・テーブル項目ではなくなったことを示すように有効フィールドを設定する。その場合、ミスキュー・テーブル６００内のその行を、新しい項目が使用することができる。

【００４８】新規命令８００のデータがキャッシュに入っているものとする。次に、新規命令８００がキャッシュ内で見つかったデータを使用して完了するものと設定される。新規命令８００が使用すると思われるアドレスを、ミスキュー・テーブル６００内のすべてのより古い項目内のアドレス情報５１０と比較する。新規命令８００のアドレスと一致するものが、ミスキュー・テーブル６００内のより古い命令に対応する項目内のアドレス情報フィールド５１０と一致する場合、それらの一致項目はアウト・オブ・オーダーとしてマークされる。アウト・オブ・オーダーは、より新しい命令が、ミスキュー・テーブルにまだ入っているより古い命令も使用するアドレスにあるデータを使用して完了するとマークされていることを意味する。アウト・オブ・オーダー・フィールド５２０は項目をアウト・オブ・オーダーとしてマークするのに使用する。

【００４９】図８に、次のクロック・サイクルで命令８００が完了した後のミスキュー・テーブル６００を示す。命令項目６２０は、完了しておらず、新規命令８００と同じ記憶場所にあるデータにアクセスするため、アウト・オブ・オーダー・フィールド５２０を使用するアウト・オブ・オーダーとしてマークされていることに留意されたい。

【００５０】新規命令８００がキャッシュからのデータを待ち続けなければならない場合、図９に示すようにミスキュー・テーブルに項目６１０として入れられ、完了可能になるまでそこにとどまる。図９には、その間に、ミスキュー・テーブル６００に他のいくつかの命令項目が追加されている様子も図示されている。後の時点で、データ・キャッシュに、新規項目６１０には使用可能になっているが一致項目６２０には使用可能になっていないデータが入っている。したがって、新規命令８００が完了した。

【００５１】新規命令８００が完了すると、プロセッサは命令８００より古い命令に対応するミスキュー・テーブル内の項目、すなわち対応する項目６１０より上のすべての項目を走査する。プロセッサは、より古い項目を探索して新規命令項目のアドレス・フィールド６５０と一致するアドレスがないか調べる。新規命令項目６１０と同じアドレスに行く、より古い命令項目が見つかった場合、図１０に示すようにその古い命令項目がアウト・オブ・オーダーとしてマークされる。

【００５２】新規命令８００が完了すると、ミスキュー６００内のその項目が削除される。好ましい実施形態では、すべての他の項目が上に移動し、したがって図９及び図１０に示すようにテーブルのタイミング順序が保持される。

【００５３】任意の時点で命令のデータが変わった場合、その項目はそのことを示さなければならない。したがって、ミスキュー・テーブル６００内に命令項目がある間にスヌープまたはその他のデータ変更の表示が行われると、対応する命令の変更標識５３０が潜在的な問題を示すように設定される。プロセッサは項目を走査し、データ変更時に同じアドレスを示すアドレスを有する各項目を、変更されているものとしてマークする。

【００５４】データ・コヒーレンシが侵害された可能性がある場合、アウト・オブ・オーダー標識と変更標識の両方によって、それらの事象が発生したことが示される。すなわち、より古い命令より先に完了するものとして設定されていたより新しい命令に、より古い命令より古いデータが関連づけられている可能性がある。これによっておそらくデータ・コヒーレンシが侵害されることになる。

【００５５】図１１、図１２、図１３、及び図１４に、この問題を検出するシーケンスを示す。ミスキュー・テーブル６００は、それぞれ命令１２１０、１２２０、及び１２３０に対応する初期項目１１１０、１１２０、及び１１３０を有する。記憶場所ａに行く新規命令１２４

０が発行され、そのデータがキャッシュ内で見つかる。すると、図１２で項目１１１０はそれより前に新規命令１２４０が完了すると、アウト・オブ・オーダーとしてマークされる。図１３で、記憶場所ａが変更されたことを示すスヌープ要求１１４０が行われる。ミスキュー・テーブル６００に問い合わせると、項目１１１０が変更済みとしてマークされている。したがって、項目１１１０はａから来るデータについてデータ・コヒーレンシ問題を示す。

【００５６】新規命令１２４０は、ただちにクリアされる場合でもミスキュー・テーブル６００に項目１１５０を加えたことに留意されたい。新規命令１２４０がキャッシュでミスした場合、加えられた項目１１５０はミスキュー・テーブル６００内に残ることに留意されたい。また、項目１１５０はミスの後に作成することもできることにも留意されたい。重要なことは、新規命令１２４０のオペランド・アドレスがミスキュー・テーブル内の項目と比較され、新規命令１２４０がより古い項目１１１０、１１２０、及び１１３０に対応する命令よりも前に完了する場合に、一致するものがあればそれをマークする必要があることである。

【００５７】項目１１１０の場合のように、命令項目が変更され、アウト・オブ・オーダーであることが示された場合、より古い命令項目にアウト・オブ・オーダーのマークを付けさせるそのより新しい命令を実行することはできない。好ましい実施形態では、項目１１１０に対応する命令が完了すると、ミスキュー・テーブルはその項目に対応する命令に問題が発生したことを完了論理回路に報告する。項目１１１０に対応する命令は完了し、実行することができる。しかし、その開始項目１１１０の後に続くすべての命令、すなわち命令１２２０、１２３０、及び１２４０と、完了して項目１１１０に対応する命令の後に実行するように設定された他の命令は取り消され、実行プロセス全体を再び通過するようにリセットされる。

【００５８】本発明の好ましい実施形態では、アドレスはセット細分性とのみ比較される。したがって、アドレスはダブル・ワード境界のみと比較されて、潜在的な問題を示す。しかし、実アドレス及びその他の細分性でも比較可能であることに留意されたい。

【００５９】本発明の好ましい実施形態では、エラー状況が検出された後、問題を示す命令の後のすべての命令がフラッシュされ、実行のためにリセットされる。したがって、残りの命令のフラッシュでデータ・コヒーレンシが保持される。

【００６０】まとめとして、本発明の構成に関して以下の事項を開示する。

【００６１】（１）命令のアウト・オブ・オーダー実行のための装置であって、

ａ）各項目がコンピュータ・マイクロプロセッサ内の命令に対応する命令データの項目を保持するミスキュー・テーブルを含み、前記命令データは、

ｉ）命令のアドレス情報と、

ii）他の命令を基準にした前記対応する命令の順序を示す命令の順序情報と、

iii）アドレス情報フィールドに対応するアドレスにあるデータを使用するより新しい命令が現行項目より前に完了したことを示すアウト・オブ・オーダー情報と、

iv）前記アドレス情報フィールドに対応する前記アドレスにあるデータが変更された可能性を示す、命令のデータ変更情報とを含み、前記装置は、

ｂ）比較された項目内のアドレス情報フィールドに対応するアドレスにあるデータを使用する、より新しい命令が前記比較された項目より前に完了した場合、前記比較された項目のアウト・オブ・オーダー情報フィールドが設定される、前記ミスキュー・テーブル内の項目に前記アウト・オブ・オーダー情報フィールドを設定するアウト・オブ・オーダー比較機構と、

ｃ）前記ミスキュー・テーブル内の前記アドレス情報フィールド内のアドレス情報を、変更された可能性のあるアドレスと比較する変更比較機構とを含み、前記変更されたアドレスが比較された前記命令項目内の前記アドレス情報と一致する場合、前記項目内の前記変更フィールドが、前記アドレスにある変更されたデータを示すようにマークされる装置。

（２）アウト・オブ・オーダー命令と変更されたデータとを示す命令項目に対応する命令のすべての後続命令を取り消す、上記（１）に記載の装置。

（３）前記アウト・オブ・オーダー比較機構が項目内のすべてのデータ・アドレス情報を比較して前記アドレスが一致するかどうかを判断する、上記（１）に記載の装置。

（４）前記アウト・オブ・オーダー比較機構が、項目内の前記データ・アドレス情報の一部を比較して、前記アドレスが一致するかどうかを判断する、上記（１）に記載の装置。

（５）データ・コヒーレンシ侵害を生じさせる可能性のあるマイクロプロセッサ内のアウト・オブ・オーダー命令を検出する方法であって、

ａ）データ・アドレスを有する前記マイクロプロセッサ上で実行する新規命令を作成するステップと、

ｂ）前記新規命令が実行されるように設定されている場合、

ｉ）前記命令のデータ・アドレスをミスキュー・テーブル内の、前の命令に対応し、アドレス情報と命令順序情報とアウト・オブ・オーダー標識と変更データ標識とを含む既存の命令項目と比較し、

ii）前記新規命令のデータ・アドレスがミスキュー・テーブル内の項目のアドレス情報と一致する場合、前記ミスキュー・テーブル内の前記一致項目をアウト・オブ・

オーダー命令にするステップと、
ｃ）前記新規命令が実行されるように設定されていない場合、前記ミスキュー・テーブル内の前記新規命令の項目を作成し、前記新規命令のデータ・アドレスが前記新規命令項目のアドレス情報に入れられ、前記命令項目の順序情報に前記命令の順序に関する情報が入れられる方法。
（６）ｄ）前記ミスキュー・テーブルを絶えず走査して、実行するように設定されている命令に対応する項目がないか調べるステップと、
ｅ）前記ミスキュー・テーブル内の項目に対応する命令が実行されるように設定されている場合、
ｉ）実行するように設定されている前記命令に対応する前記項目のアドレス情報を、実行するように設定されている前記命令より古い命令に対応する前記ミスキュー・テーブル内の項目のアドレス情報と比較するステップと、
ii）前記ミスキュー・テーブル内に、実行するように設定されている前記命令より古い命令に対応する項目が見つかった場合、及び実行するように設定されている前記命令に対応する前記項目のアドレス情報が前記より古い命令に対応する前記項目内のアドレス情報と一致する場合、前記より古い命令に対応する前記項目をアウト・オブ・オーダーとしてマークするステップと、
iii）実行するように設定されている前記命令に対応する前記項目を前記ミスキュー・テーブルから除去するステップとをさらに含む、上記（５）に記載の方法。
（７）ｆ）データ変更事象がないか絶えず走査するステップと、
ｇ）データ変更事象が発生した場合、
ｉ）前記変更されたデータのアドレスを前記ミスキュー・テーブルにブロードキャストするステップと、
ii）前記変更されたデータのアドレスを前記ミスキュー・テーブルの項目内のアドレス情報と比較するステップと、
iii）前記変更されたデータのアドレスと前記ミスキュー・テーブル内の項目のアドレスが一致する場合、前記一致するアドレスを有する前記ミスキュー・テーブル内の項目を変更されたものとしてマークするステップとをさらに含む、上記（６）に記載の方法。

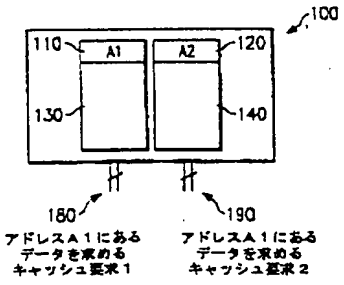
【図面の簡単な説明】

【図１】キャッシュ・バンクの１つにある同じデータにアクセスを試みる２つの命令を示す、インタリーブ・キャッシュの図である。
【図２】図１のキャッシュ内のより古い命令の前にデータにアクセスする、より新しい命令と、アウト・オブ・オーダー完了がどのように行われるかを示す図である。
【図３】スーパースカラ・プロセッサを示すブロック図である。
【図４】スーパースカラ・プロセッサ内のロード回路を示すブロック図である。
【図５】本発明の好ましい実施形態によるミスキュー・テーブルを示す図である。
【図６】各フィールドを示すミスキュー・テーブル内の項目の図である。
【図７】本発明の一実施形態によってアウト・オブ・オーダー命令完了を検出する様子を示す図である。
【図８】本発明の一実施形態によってアウト・オブ・オーダー命令完了を検出する様子を示す図である。
【図９】本発明の一実施形態によってアウト・オブ・オーダー命令完了を検出する様子を示す図である。
【図１０】本発明の一実施形態によってアウト・オブ・オーダー命令完了を検出する様子を示す図である。
【図１１】本発明の一実施形態がどのように機能するかを示す図である。
【図１２】本発明の一実施形態がどのように機能するかを示す図である。
【図１３】本発明の一実施形態がどのように機能するかを示す図である。
【図１４】本発明の一実施形態がどのように機能するかを示す図である。

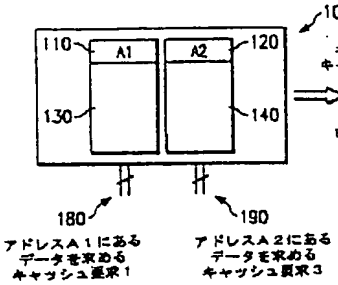
【符号の説明】

１０　プロセッサ・システム
１１　システム・バス
１２　バス・インタフェース・ユニット
１４　命令キャッシュ
１６　データ・キャッシュ
１８　シーケンサ・ユニット
２０　分岐ユニット
２２　固定小数点ユニットＡ
２４　固定小数点ユニットＢ
２６　複素固定小数点ユニット
２８　ロード／ストア・ユニット
３０　浮動小数点ユニット
３２　汎用アーキテクチャ・レジスタ
３４　固定小数点リネーム・バッファ
３６　浮動小数点アーキテクチャ・レジスタ
３８　浮動小数点リネーム・バッファ
３９　システム・メモリ
４０　特殊目的レジスタ
４６　ディスパッチ・ユニット
２０２　加算器
２０４　データ・ユニット
２０６　キャッシュ
２０８　不整列／ビジー・ラッチ
２１０　フォーマッタ

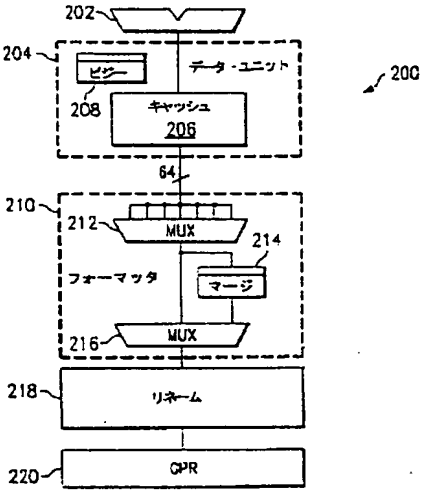
【図１】



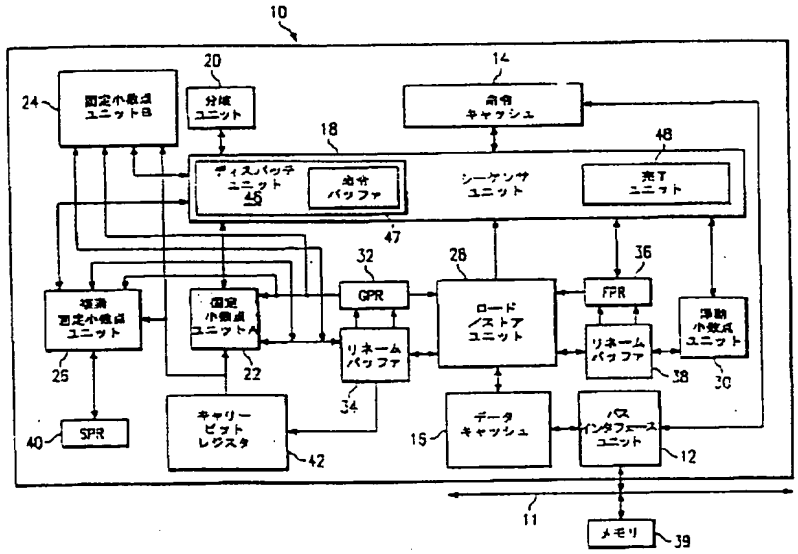
【図２】



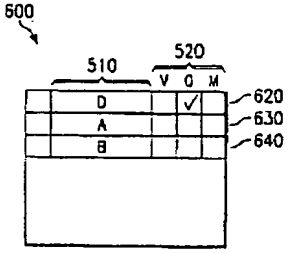
【図４】



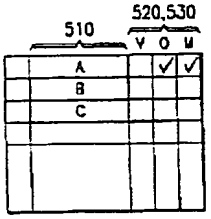
【図３】



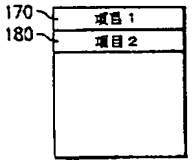
【図８】



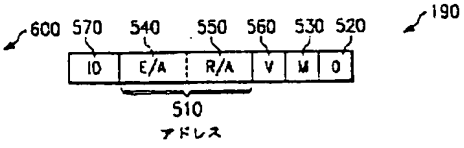
【図１４】



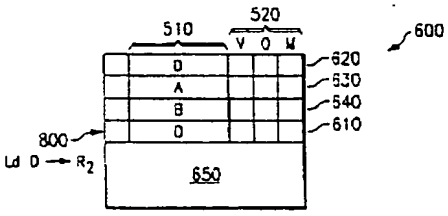
【図５】



【図６】

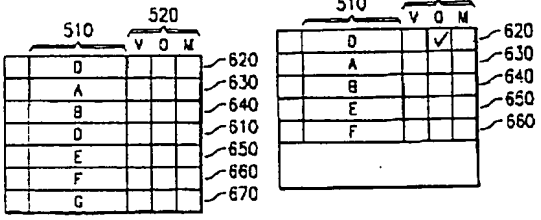


【図７】

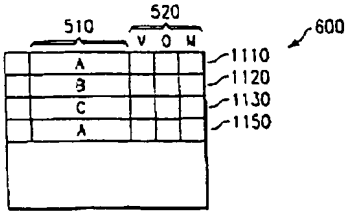


【図１０】

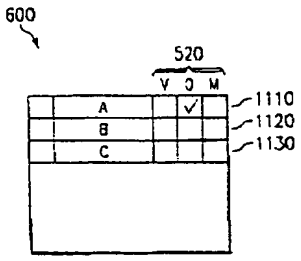
【図９】



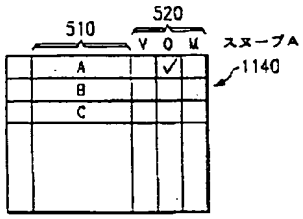
【図１１】



【図１２】



【図１３】



フロントページの続き

(72)発明者　デーヴィッド・スコット・レイ
アメリカ合衆国78628　テキサス州ジョージタウン　ヤング・ランチ・ロード　700

(72)発明者　ケビン・アーサー・キャロット
アメリカ合衆国78660　テキサス州フルーガービル　ヘイワース・コーブ　17907

(72)発明者　バリー・デュアン・ウイリアムソン
アメリカ合衆国78681　テキサス州ラウンドロック　リバーローン・ドライブ　807